Converting NWSRFS Mods to FEWS xml files

Background:

There will be a period of time where the RFCs will execute FEWS and NWSRFS in parallel (parallelOps). During parallelOps there is a need to automatically reproduce mods created in NWSRFS in FEWS. This will avoid the need to create mods in both systems. Even more important is the ability to reproduce mods in FEWS the RFCs are not responsible for (e.g. SSARREG from external users and WECHNG from external system).

The goal of this work is to create a utility that will automatically convert mods created/loaded in IFP to FEWS xml files. The created xml files will then be available for ingest into FEWS using an import workflow.

Note: Currently, SSARREG and WECHNG are the only mods supported

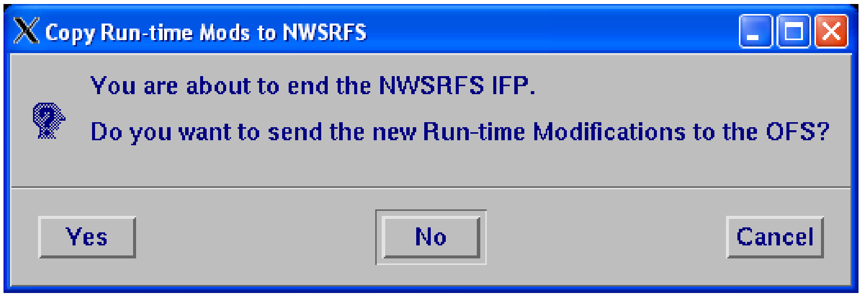
Deliverables:

1. ohdfewsadapter.jar (update to current version – RELEASE-2.X.X)
2. ohdcommon.jar (new jar file - RELEASE-2.X.X)
3. parse_mods_by_segment_for_translator (new C executable - RELEASE-2.X.X)
4. translate_rfs_mods_to_fews_script (new IFP script)

Note: Items 1 – 3 are part of standard CHPS release

How it Works

When running IFP, the utility to convert NWSRFS mods to FEWS xml files is executed as a background process when a user selects “Yes” to the question “Do you want to send the new Run-time Modifications to the OFS?



You are about to end the NWSRFS IFP.

Do you want to send the new Run-time Modifications to the OFS?

Today, all RFCs have a baseline script (named fcst_script) in the ifp scripts directory ($ifps_scripts_dir) that executes some user defined jobs when mods are sent back to OFS. This script should be edited to add the following lines (highlighted in yellow).

```
#!/bin/ksh
#read fgroup name
fgroup=$1

echo command line args are: $1 $2 $3 $4

#set variable for ofs_input directory
INPUT_DIR=`get_apps_defaults ofs_input`
OUTPUT_DIR=`get_apps_defaults ofs_output`

#============================================================================
printf "\n\n==== Start translating mods from RFS text to FEWS xml for forecast group $fgroup ====\n"
IFP_SCRIPTS_DIR=`get_apps_defaults ifp_scripts_dir`

# chps_ohd_bin_dir is the directory with the CHPS OHD binaries and jars
CHPS_OHD_BIN_DIR=`get_apps_defaults chps_ohd_bin_dir`
# chps_mods_import_dir is where the translated xml files are put by translator; for ingest into FEWS
CHPS_MODS_IMPORT_DIR=`get_apps_defaults chps_mods_import_dir`

$IFP_SCRIPTS_DIR/translate_rfs_mods_to_fews_script $fgroup $CHPS_OHD_BIN_DIR
$CHPS_MODS_IMPORT_DIR
exitValue=$?

if [[ $exitValue -eq 0 ]];then
   printf "==== Finished translating mods from RFS text to FEWS xml for forecast group $fgroup ====\n\n"
else
   printf "==== ERROR translating mods from RFS text to FEWS xml for forecast group $fgroup ====\n\n"
fi
#============================================================================

ofs -p fcst -i v5$fgroup -u oper -o v5$fgroup
```

Sample fcst_script

The highlighted changes reference 3 new things:

1. translate_rfs_mods_to_fews_script (a new IFP script)
2. chps_ohd_bin_dir (a new Apps_defaults token)
3. chps_mods_import_dir (a new Apps_defaults token)

The new script *translate_rfs_mods_to_fews_script*, which should be in the directory pointed to by the token $ifp_scripts_dir, :

1. executes a C program (parse_mods_by_segment_for_translator); this program puts all NWSRFS mod text files sent back to OFS (for the current Forecast Group) into a temporary directory
2. translates the NWSRFS mod text files into a pair of xml files (timeseries and meta-data)

The new Apps_defaults tokens should be added to the appropriate Apps_defaults files. **There is not a default value in the national Apps_defaults file.**

1. *chps_ohd_bin_dir* is the directory with the CHPS OHD binaries and jars
2. *chps_mods_import_dir* is where the translated xml files are put by the translator; for ingest into FEWS (same as IMPORT_FOLDER_MODS, defined in sa_global.properties)

<u>Configuring the Script</u>

The new script *translate_rfs_mods_to_fews_script* has a portion at the top for site-specific configuration (shown below)

```
# ==================== start: needs to be configured by rfcs =====================
# uncomment the appropriate value for mod units (i.e. same as technique MODUNITS); default is ENGLISH
#MOD_UNITS=ENGLISH
#MOD_UNITS=METRIC
# uncomment the appropriate value for mod time zone code (i.e. same as technique MODTZC); default is Z
#MOD_TZC=Z
#MOD_TZC=EST
#MOD_TZC=EDT
#MOD_TZC=CST
#MOD_TZC=CDT
#MOD_TZC=MST
#MOD_TZC=MDT
#MOD_TZC=PST
#MOD_TZC=PDT
# uncomment a value for DEBUG_FLAG; default is “false”
#DEBUG_FLAG=true
#DEBUG_FLAG=false
# =================== end: needs to be configured by rfcs =========================
```

Top of translate_rfs_mods_to_fews_sciprt

There are three pieces of optional information to configure.

1. MOD_UNITS **(optional)** = a property used to represent the current setting for the NWSRFS technique MODUNITS; this is used to identify the system of units (ENGLISH/METRIC) for mods. You can specify this value by removing the comment character “#” from one of the options. If none of the options are uncommented “ENGLISH” is used.
2. MOD_TZC **(optional)** = a property used to represent the current setting for the NWSRFS technique MODTZC; this is used to identify the time zone code for mod dates when not explicitly identified in the mod. You can specify this value by removing the comment character “#” from one of the options. If none of the options are uncommented “Z” is used.
3. DEBUG_FLAG **(optional)** = a property used to control whether debug level messages are logged in the output diagnostic file. You can specify this value by removing the comment character “#” from one of the options. If none of the options are uncommented “false” is used.

<u>Output</u>

Executing the script results in:

1. One or more timeseries xml files whose filename consists of the mod name, location id, and mod dates.

   For example:
   TimeSeries_WECHNG_ENVI1U_12282009120000_12282009120000_12282009120000.xml
   is the file for a WECHNG mod for segment ENVI1U starting, ending, and valid on 12/28/2009 12:00:00

2. A metadata xml file (one for each time series xml file) containing metadata about the mod. The naming convention is the same as the time series xml file with the only difference being the prefix “MetaData”.

   For example:
   MetaData_WECHNG_ENVI1U _12282009120000_12282009120000_12282009120000.xml

3. A diagnostic file (one per forecast group) with messages used to debug the translation program. This file is not ingested by FEWS. Subsequent runs of the translation script for the same forecast group will append to the same file. The script has logic to remove diagnostic xml after one week.

   For example, diag_ABCD.xml, contains all diagnostic messages for translating mods for forecast group “ABCD”.